COL 380 : A3

Sidharth Agarwal - 2019CS50661

March 2, 2022

Approaches tried

So these are the optimizations/approaches that I tried but are not present in the final submission.

1. **Broadcast Input:** So since usually only one of the mpi nodes have the input file, so the input needs to be broadcasted to every other node. But in our case we were told that input file is available to every node.
2. **Dividing work per node:** I also thought of dividing number of walks among ranks for each node, rather than the node itself. After that we would sum all the influence score to get final recommendations. But this will not work in our case because each rank will have same seed for randomizer. Hence the sequence of walks generated by all the ranks will be same, and thereby doing duplicate computation across the nodes.
3. **Send & Recv:** So I thought that we can divide work for different nodes in different MPI ranks. And then in the end rank=0 can write in the file by receiving information from each node. But since every login node have access to the same memory space, and we know the offset at which will write for every node. There is no need of send and recv, we can directly write at the write offset.
4. **Graph Partitioning:** Now I also divided the work of nodes based $(num\_nodes/size) * (rank)$ and $(num\_nodes/size) * (rank + 1)$, partitions, but soon realized that this is not equally distributing work among all the rank, and leaving a lot of computation to do on rank=0.

Approach Used

So in my submssion, I didn't use any send and recv and directly wrote into the output file from each rank. I also partitioned the graph based on the modulus of their index, i.e $node\_num\%size = i$ goes to rank = i, this clearly reduced the time computation by more equally distributing the work among ranks. We can improve this by using some sort of advanced algorithm, which paritions graph in more uniform way(like sort the nodes on basis of length of its edge list and then assign to MPI ranks based on their modulus).

Experiment

So in all the experiments below use $num\_steps = 250$, $num\_walks = 30$, $probability\_factor = 0.1$, $num\_rec = 20$, $seed = 369$

File 1

So this file had $num\_nodes = 8717$ and $num\_edges = 31525$

MPI_Size	Time (in s)
1	23.813
2	12.170
4	6.328
8	3.400
16	3.422

File 2

So this file had $num\_nodes = 81867$ and $num\_edges = 545671$

MPI_Size	Time (in s)
1	827.440
2	405.454
4	216.65
8	130.60
16	65.993

File 3

So this file had $num\_nodes = 82168$ and $num\_edges = 870161$

MPI_Size	Time (in s)
1	2573.442
2	1266.661
4	743.531
8	383.75
16	172.990